

A simple method for identifying code coverage percentage analysis

Patent number: CN1630052
Publication date: 2005-06-22
Inventor: SUN JIE (CN); LI WEIMIN (CN); HUA HAIHONG (CN)
Applicant: NANSHANBRIDGE CO LTD (CN)
Classification:
- International: *G06F11/00; G06K5/00; H01L21/00; H01L21/66; G06F11/00; G06K5/00; H01L21/00; H01L21/66; (IPC1-7): H01L21/66; G06F11/00; G06K5/00; H01L21/00*
- european:
Application number: CN200310104038 20031218
Priority number(s): CN200310104038 20031218

Report a data error here

Abstract of CN1630052

This invention discloses a method for establishing verification code coverage percentage in chip verification, which contains marking to branch of logic code, simulation to operation code, recording passed logic branch, judging simulation, and statistics to total passed logic branch. Said invention realizes the analysis to verification code coverage percentage and can find in time the hidden logic errors in complex logic ultra large scale IC to keep it in normal operation.

Data supplied from the *esp@cenet* database - Worldwide

THIS PAGE BLANK (USPTO)

[19] 中华人民共和国国家知识产权局

[51] Int. Cl⁷

H01L 21/66

H01L 21/00

G06F 11/00

G06K 5/00



[12] 发明专利申请公开说明书

[21] 申请号 200310104038.5

[43] 公开日 2005 年 6 月 22 日

[11] 公开号 CN 1630052A

[22] 申请日 2003.12.18

[21] 申请号 200310104038.5

[71] 申请人 四川南山之桥微电子有限公司

地址 611731 四川省成都市高新技术(西
区)创业中心 C241

[72] 发明人 孙 杰 李为民 华海红

[74] 专利代理机构 成都天元专利事务所

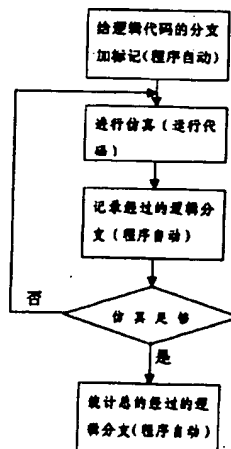
代理人 张 新

权利要求书 2 页 说明书 4 页 附图 1 页

[54] 发明名称 一种验证代码覆盖率分析的简单方法

[57] 摘要

本发明公开了一种芯片验证中建立验证代码覆盖率的方法,包括以下步骤:给逻辑代码的分支加标记、对运行代码进行仿真、记录经过的逻辑分支、仿真足够程度的判断、统计总的经过的逻辑分支,本发明实现了验证代码覆盖率的分析,简单直接,能够及时发现有复杂逻辑的超大规模集成电路中发现隐藏的逻辑错误,从而保证有复杂逻辑的超大规模集成电路的正常运行。



ISSN 1008-4274

1、一种验证代码覆盖率分析的简单方法，其特征在于：包括以下步骤：给逻辑代码的分支加标记、对运行代码进行仿真、记录经过的逻辑分支、仿真足够程度的判断、统计总的经过的逻辑分支。

2、根据权利要求1所述的一种验证代码覆盖率分析的简单方法，其特征在于：所述的具体步骤如下：

A、根据验证代码的所使用的语言特点，将代码分解成为语句块，每个语句块构成是一个逻辑分支；

B、读取验证代码文件，在语句块中自动加入代码，并自动加入的代码用数字递增方式把语句块标记出来；

C、在一次仿真运行的过程中，将覆盖过的逻辑的数字标号都输出到临时文件中存储；

D、分析该临时文件，将覆盖过的逻辑在代码运行时自动记录下来，并保存到分析文件中存储，同时删除临时文件；

E、在以后的仿真中，新覆盖的逻辑分支，同样保存在该文件中，记录整个仿真过程中覆盖到的逻辑分支；分析临时文件，将覆盖过的逻辑在代码运行时自动记录下来，并保存到分析文件中存储，同时自动删除临时文件；

F、仿真足够程度的判断，仿真足够后，进行总的经过的；逻辑分支的统计，或在仿真不够时，重新对运行代码进行仿真并记录经过的逻辑分支，直至仿真足够。

3、根据权利要求1或2所述的一种验证代码覆盖率分析的简单方法，其特征在于：所述的给逻辑代码的分支加标记是采用程序自动控制来对代码添加标记。

4、根据权利要求1或2所述的一种验证代码覆盖率分析的简单方法，其特征在于：所述的给逻辑代码的分支加标记、记录经过的逻辑分支和统计总的经过的逻辑分支的自动添加、自动记录、自动统计在一个程序中有不同的命令参数区分。

5、根据权利要求1或2所述的一种验证代码覆盖率分析的简单方法，其特征在于：所述的给逻辑代码的分支加标记的程序实现中有把已添加的逻辑分支数字标记自动删除的命令。

一种验证代码覆盖率分析的简单方法

技术领域

本发明涉及一种芯片验证中建立验证代码覆盖率的方法。

背景技术

对于有复杂逻辑的超大规模集成电路，其验证代码的参考模型对应的逻辑也会非常复杂，逻辑流程会有很多的分支，循环，跳转等，对于不同的数据包，只可能从一个逻辑入口进入，经过某条逻辑路径到达出口，所以对于一个数据包的处理过程，不可能覆盖所有的逻辑分支，而只能覆盖其中的很小一部分，当大量不同的数据包经过这些逻辑处理时，总的经过的逻辑分支会增加，这些经过了的逻辑就被称为被覆盖了的逻辑，没有经过的逻辑就是未被覆盖的逻辑。被覆盖了的逻辑分支与总的逻辑分支的比率就是代码覆盖率。其中，行覆盖率是最基本的覆盖率，即对每以行都可以运行到。

一个已经设计好的逻辑代码中所有的逻辑，都是期望能覆盖的，如果一个永远也不可能覆盖的逻辑出现的代码里，那就应该是个错误。在通常情况下，超大规模集成电路的逻辑都非常复杂，以至有可能在运行了大量数据以后还不能覆盖所有的逻辑分支。这时就需要针对未覆盖的逻辑构造特殊的数据包，来覆盖该逻辑。所以，逻辑覆盖率分析是 ASIC 验证中非常重要的一环。

目前，RTL 代码有很多工具可以分析代码覆盖率，但尚未有能够直接简单的方法来验证代码覆盖率的分析，特别是行覆盖率，也就无法在有复杂逻辑的超大规模集成电路中发现隐藏的逻辑错误，从而影响到有复杂逻辑的超大规模集成电路的正常运行。

发明内容：

本发明目的是旨在提供一种简单的验证代码覆盖率的简单方法，并使用该方法，开发出一个代码覆盖率分析工具即软件。这种方法简单直接，能够及时发现有复杂逻辑的超大规模集成电路中发现隐藏的逻辑错误，从而保证有复杂逻辑的超大规模集成电路的正常运行。

本发明具体的技术方案如下：

一种验证代码覆盖率分析的简单方法，其特征在于：包括以下步骤：给逻辑代码的分支加标记、对运行代码进行仿真、记录经过的逻辑分支、仿真足够程度的判断、统计总的经过的逻辑分支。

本发明所述的具体步骤如下：

A、根据验证代码的所使用的语言特点，将代码分解成为语句块，每个语句块构成是一个逻辑分支；

B、读取验证代码文件，在语句块中自动加入代码，并自动加入的代码用数字递增方式把语句块标记出来；

C、在一次仿真运行的过程中，将覆盖过的逻辑的数字标号都输出到临时文件中存储；

D、分析该临时文件，将覆盖过的逻辑在代码运行时自动记录下来，并保存到分析文件中存储，同时自动删除临时文件；

E、在以后的仿真中，新覆盖的逻辑分支，同样保存在该临时文件中，记录整个仿真过程中覆盖到的逻辑分支，分析临时文件，将覆盖过的逻辑在代码运行时自动记录下来，并保存到分析文件中存储，同时自动删除临时文件；

F、仿真足够程度的判断，仿真足够后，进行总的经过的逻辑分支的统计，或在仿真不够时，重新对运行代码进行仿真并记录经过的逻辑分支，直至仿真足够。

本发明所述的给逻辑代码的分支加标记是采用程序自动控制来对代码添加标记。

本发明所述的给逻辑代码的分支加标记、记录经过的逻辑分支和统计总的经过的逻辑分支的自动添加、自动记录、自动统计在一个程序中有不同的命令参数区分。

本发明所述的给逻辑代码的分支加标记的程序实现中有把已添加的逻辑分支数字标记自动删除的命令。由于代码可能不断更新，所以有把已添加的逻辑分支数字标记自动删除的能力，能方便代码更新后重新对逻辑分支依次排序。

本发明还可以用于设计代码和验证代码的逻辑分支不一致时，

快速定位检出逻辑分支不一致的地方和开始起点，即能通过显示或打印等，快速比较设计逻辑和验证逻辑的流程。

本发明实现了验证代码覆盖率的分析，简单直接，能够及时发现有复杂逻辑的超大规模集成电路中发现隐藏的逻辑错误，从而保证有复杂逻辑的超大规模集成电路的正常运行。

附图及其说明：

图1 本发明的逻辑方框图

具体实施方式：

一种验证代码覆盖率分析的简单方法，其特征在于：包括以下步骤：给逻辑代码的分支加标记、对运行代码进行仿真、记录经过的逻辑分支、仿真足够程度的判断、统计总的经过的逻辑分支。

本发明所述的具体步骤如下：

A、根据验证代码的所使用的语言特点，将代码分解成为语句块，每个语句块构成是一个逻辑分支。比如，C代码在两个大括号{}中间的语句是一个语句块。

B、读取验证代码文件，在语句块中自动加入代码，并自动加入的代码用数字递增方式把语句块标记出来；比如：一种语言，比如 Perl，开发一个软件工具，该工具有以下功能：1、能读取验证代码文件；2、能在语句块中自动加入代码，3、自动加入的代码可以用数字递增的把语句块标记出来，这为数字标号。

C、在一次仿真运行的过程中，将覆盖过的逻辑的数字标号都输出到临时文件中存储；

D、分析该临时文件，将覆盖过的逻辑在代码运行时自动记录下来，并保存到分析文件中存储，同时删除临时文件；比如：一种语言，如 Perl，开发一个软件工具，分析该文件，将覆盖过的逻辑在代码运行时自动记录下来，并保存到分析文件中存储；

E、在以后的仿真中，新覆盖的逻辑分支，同样保存在临时文件中存储，记录整个仿真过程中覆盖到的逻辑分支；分析临时文件，将覆盖过的逻辑在代码运行时自动记录下来，并保存到分析文件中存储，同时自动删除临时文件；

F、仿真足够程度的判断，仿真足够后，进行总的经过的；逻辑分支的统计，或在仿真不够时，重新对运行代码进行仿真并记录经过的逻辑分支，直至仿真足够。

本发明所述的给逻辑代码的分支加标记是采用程序自动控制来对代码添加标记。

本发明所述的给逻辑代码的分支加标记、记录经过的逻辑分支和统计总的经过的逻辑分支的自动添加、自动记录、自动统计在一个程序中有不同的命令参数区分。

本发明所述的给逻辑代码的分支加标记的程序实现中有把已添加的逻辑分支数字标记自动删除的命令。由于代码可能不断更新，所以有把已添加的逻辑分支数字标记自动删除的能力，能方便代码更新后重新对逻辑分支依次排序。

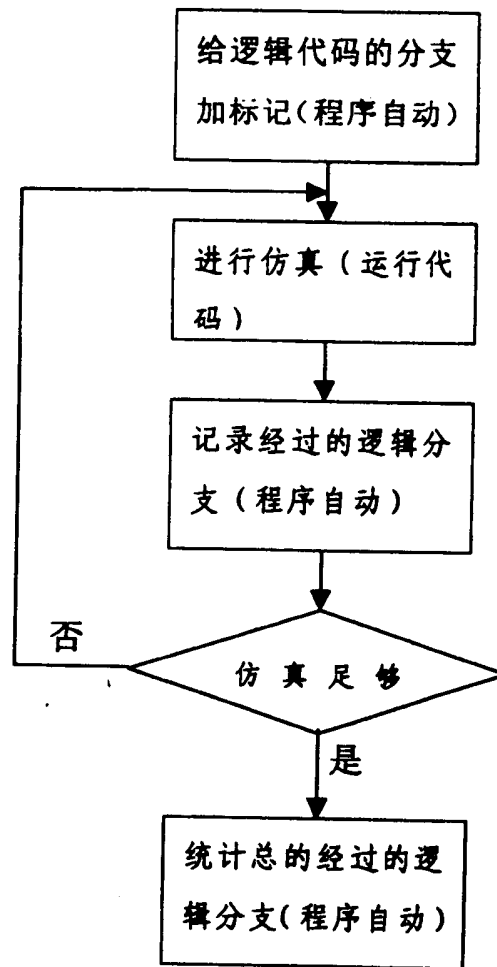


图 1